RESEARCH ARTICLE OPEN ACCESS

Vol. 99, No. 1, Month 2099, pp. 1~1x
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v99.i1.pp1-1x

IMACO: A Security-Floor-Constrained Adaptive Cryptography Framework for Constrained CoAP-Class IoT

Adnan H. Al-Helali[1], Judy Ammar Jaradat[2]

[1,2] Department of Cybersecurity, College of Science and Information Technology, Irbid National University, Irbid, Jordan

Article Info

Article history:

Received September 2026
Revised month dd, yyyy
Accepted month dd, yyyy

Keywords:

IoT
Crypto-agility; downgrade resistance; security protocols; authenticated profile switching; AEAD; post-quantum migration; software prototype

ABSTRACT

Constrained CoAP-class Internet of Things (IoT) devices, including deployments protected by OSCORE, must maintain authenticated protection while operating under changing energy, memory, latency, and packet-overhead limits. Controller-driven adaptation may improve efficiency, but it can also create a security-downgrade path when a gateway influences the selection decision. This study proposes IMACO, a security-floor-constrained adaptive cryptography framework that keeps final authority on the endpoint. The method stores a small, signed catalog of approved authenticated-encryption profiles, assigns each profile a security label and predicted resource costs, filters candidates against a locally enforced security floor and current device budgets, and ranks only the remaining safe and feasible profiles. Gateway observations may refine cost or utility estimates but cannot modify the local floor or profile labels. Formal analysis under a network adversary that may control the gateway shows that every profile returned by the selector satisfies the device's minimum-security requirement. Cost-estimation errors may reduce efficiency or availability, but they do not authorize selection below the floor; when no safe feasible profile exists, the device uses a signed fallback or suspends the protected service. These results establish a clear separation between security enforcement and performance optimization. The framework therefore provides a defensible basis for future Contiki-NG and Cooja experiments measuring energy, latency, memory, packet overhead, and adaptation behavior.

Corresponding Author:

Judy Ammar Jaradat
Department of Cybersecurity, College of Science and Information Technology, Irbid National University, Irbid, Jordan.
Email: 202520099@inu.edu.jo

1. INTRODUCTION

Internet of Things devices include sensors, meters, wearables, controllers, and small actuators. Many of them

risk, yet the same type of device can appear in a hospital, factory, or smart building, where a changed command or an exposed measurement can affect people and physical equipment. Security is necessary even when a node has a small processor, little memory, and a battery expected to last months or years.

The hard part is not deciding to encrypt. Systems also have to replace or select algorithms as hardware support, protocol versions, and security requirements change, a property known as crypto-agility. It helps, but it opens a negotiation problem: a peer that influences the choice may try to keep an obsolete or weaker option enabled. Performance can rank options that are already acceptable, but it must not decide which security properties can be dropped.

This research uses authenticated encryption with associated data (AEAD) as its basic protection tool. AEAD provides confidentiality and integrity while authenticating selected visible headers. AES is defined in FIPS 197 [3]; GCM and CCM are standardized AEAD modes [4], [24]. RFC 8439 specifies ChaCha20-Poly1305 [5]. NIST SP 800-232 standardizes Ascon-AEAD128 for constrained devices [1], based on the Ascon design analyzed in [2].

Several standard protocols can carry these protections. CoAP was designed for constrained environments [20]. OSCORE protects a CoAP message from one endpoint to the other even when an intermediary forwards it [6]. EDHOC provides a compact authenticated key exchange that can create keying material for OSCORE [7]. DTLS 1.3 protects datagram channels [8]. Studies show that OSCORE and EDHOC can run on constrained devices, but their cost changes with the board, payload, and workload [9], [10]. That variation is what makes adaptive selection worth doing, as long as cost never overrides a required security property.

A simple adaptive design would let the gateway pick the cheapest compatible profile, but that hands the gateway too much authority. A compromised gateway could request a weak profile, replay an old configuration, or trigger repeated expensive changes. These are downgrade and availability problems, not just performance ones. Adaptive cryptography is already established work [12], [13], [14], [15], [18]; the contribution here is not a new cipher or the first adaptive framework.

The question this paper asks is simple: can a constrained endpoint use information from the gateway without letting that gateway push security below a local minimum? IoT Minimum-Assurance Cryptographic Orchestration (IMACO) changes the order of the decision. The node first drops every profile below its local floor, then ranks only what is left using measured cost and optional authenticated advice. If nothing meets the floor, it stops rather than quietly weakening protection.

The paper's contributions are an executable four-profile catalog, a deterministic floor-first selector, an authenticated two-phase transition, and a reproducible software evaluation. It also spells out the cryptographic assumptions, nonce and tag limits, compromise scope, and the amortization break-even point that bound the result. IMACO is a security-protocol and crypto-agility mechanism, not a new primitive and not a microcontroller performance evaluation.

Section 2 reviews cryptographic profiles, adaptive selection, downgrade attacks, and crypto-agility. Section 3 covers the threat model, the catalog, the selector, the security argument, the safe-switch implementation, and the measurement method. Section 4 reports the checks that were run, the cost results, the break-even analysis, the comparison, and the limitations. Section 5 concludes.

2. BACKGROUND, LITERATURE REVIEW, AND RELATED

2.1 Cryptographic profiles rather than isolated ciphers

Simply naming a cipher is not enough to describe the security of an IoT device connection. A configuration must also state how the devices in the connection authenticate, manage nonces, avoid replayed messages, establish and renew keys, and where the protection of the data begins and ends. The cipher AES-GCM encrypts the data but requires unique nonces and key management. Ascon-AEAD128 also requires a secure method of establishing cryptographic keys. For these reasons, IMACO selects profiles rather than algorithm names alone.

Each profile includes information about the AEAD algorithm, the key-establishment method, the replay policy, the rekeying rule, and on which communication layer the protection will operate. The prototype implements the following: an authenticated encryption AEAD algorithm, per-epoch keys, authenticated switching, and monotonicity of replayed messages. This profile does not implement the full OSCORE, EDHOC, or DTLS protocol but will integrate with these protocols in the future.

OSCORE security can be applied to the scenario modeled by IMACO because a gateway can forward messages without decrypting them [6]. The security context of OSCORE includes the keying material, the identifiers, the AEAD algorithm, and the sequence number state. Like OSCORE, EDHOC can establish this information through an authenticated ephemeral key exchange [7]. Published research into the protocol has revealed several security flaws in previous versions of the protocol that have been addressed [11]. IMACO will use these protocols as integration targets within the system**.**

DTLS 1.3 is another option for channel protection. It provides confidentiality, integrity, and authentication for datagram traffic, and it can cope with loss and reordering [8]. That distinction is not trivial.

OSCORE protects the application object even when it travels through an intermediary. DTLS, by contrast, protects a channel that terminates at the DTLS peer. The chosen protection layer therefore determines what a gateway can inspect, so every IMACO profile needs to state the choice plainly. OSCORE is usually the stronger fit when the gateway is not trusted. For a direct endpoint-to-endpoint connection, DTLS can be used instead.

2.2 Lightweight and standardized authenticated encryption

NIST SP 800-232 defines Ascon-AEAD128, Ascon-Hash256, and extendable-output functions for constrained devices [1]. The Ascon family uses a 320-bit permutation and has received broad public analysis [2]. It can suit platforms on which AES performs poorly, though it will not necessarily be the quickest option in every setting.

Hardware shifts the equation.

AES can be highly efficient when hardware support is present, while ChaCha20-Poly1305 is often a sensible software choice. Measurements from the actual board should steer an adaptive system; the security floor should remain distinct from the cost model. Easy to say—surprisingly easy to muddle in practice.

COSE provides algorithm identifiers for constrained protocols [19]. IMACO should enable a profile only when its algorithms and parameters have supported identifiers and tested implementations in the selected stack. Put another way, Ascon may merit testing in an experimental application profile, but without the required COSE binding and peer support, it should not be described as a standard OSCORE profile. The main catalogue thus keeps interoperable profiles apart from experimental ones.

2.3 Adaptive cryptography and context-aware security

Farooq et al. used weighted optimisation to pair several AES hardware implementations with different IoT devices [12]. Their findings suggest that throughput and resource costs can shape cryptographic choices, though a server still assigns those choices. The node, then, is not the main authority.

CALIS is more closely related to IMACO. It selects among Ascon, AES-GCM, and ChaCha20-Poly1305, while XGBoost running at an SDN controller adjusts key-rotation intervals [13]. CALIS reports practical and formal results, but depends on a trusted controller and a Linux-class testbed. IMACO goes another way: a constrained node receives recommendations from a gateway it does not fully trust.

Other work changes protection as circumstances shift. CAEM-ESAC connects contextual factors—user identity, location, access time, and device characteristics—to encryption and access-control decisions in cloud systems [14]. A reinforcement-learning approach likewise adjusts encryption and LoRa transmission settings together [15].

Kalaria et al., viewed from another angle, place context-aware access decisions at the fog layer [18]. Trust is the awkward part. A component that provides context may also sway security decisions, so these systems reveal both the value of context and the point at which it becomes risky. IMACO treats such information as advice; it therefore cannot reduce the node's local minimum. That may sound like a small distinction, but it has teeth.

2.4 Measured cost of constrained security

Measurements on real devices show that the cost of constrained security varies by platform. Gunnarsson et al. tested OSCORE on Contiki-NG devices and compared it with CoAP over DTLS [9]. Hristozov et al. measured compact OSCORE and EDHOC implementations on constrained microcontrollers [10]. Group OSCORE experiments found that signatures and hardware support can substantially alter time and energy use [16].

There is no universal ranking. Across these studies, standard end-to-end protection appears practical, yet every board needs its own measurements. Developers also need to refresh cost estimates occasionally. Platform differences are decisive—frustratingly, there is no real shortcut around them.

In light of this evidence, IMACO is best understood as a safety layer around adaptive selection. The node constructs the safe set locally, which means an assisting gateway cannot turn advice into a downgrade; an authenticated transition blocks that path. The following section examines this narrow claim through executable cryptography and explicit adversarial cases. It does not assume that adaptation improves performance on every platform, since that usually depends on the device.

3. MATERIALS AND METHOD

3.1 System and threat model

The prototype consists of three logical roles: an IoT node, an advisory gateway, and an application endpoint. The node is the trust root for its minimal-security profile. It stores the data-class-to-floor mapping table, the approved profile catalog, the policy digest, the active epoch, active profile, and packet sequence number. The gateway may suggest a profile, but has no cryptographic keys and cannot alter the local policy. The node and endpoint protect the confidentiality and integrity of application data with the active AEAD context, establishing a secure end-to-end communication channel.

The network attacker can observe, inject, modify, replay, reorder, delay, or drop messages. A compromised gateway can advise a below-floor profile or replay a previous switch record. The implemented tests include downgrade advice, switch replay, profile-identifier fuzzing, ciphertext tampering, and protected-packet replay. Physical extraction of the key, side-channel observation, denial-of-service attacks, compromised endpoints, and implementation-level hardware faults in a cryptographic primitive are beyond the threat model's scope.

The design goals require protecting the application data's confidentiality and integrity, authenticating key-establishment messages if needed, detecting replayed messages, downgrades, and enforcing the local floor. Availability is not a concern, but it is prioritized lower than others. The resource-constrained node may refuse to process a message protected with insufficient security, thus delaying or disrupting communication on a per-packet basis. Such behavior is acceptable for the confidentiality-first architecture, where a medical emergency alert must not be canceled due to inadequate transport-layer encryption.

This design philosophy is visualized in figure 1, which describes the system as a combination of executable roles, trust boundaries, and end-to-end–protected paths.

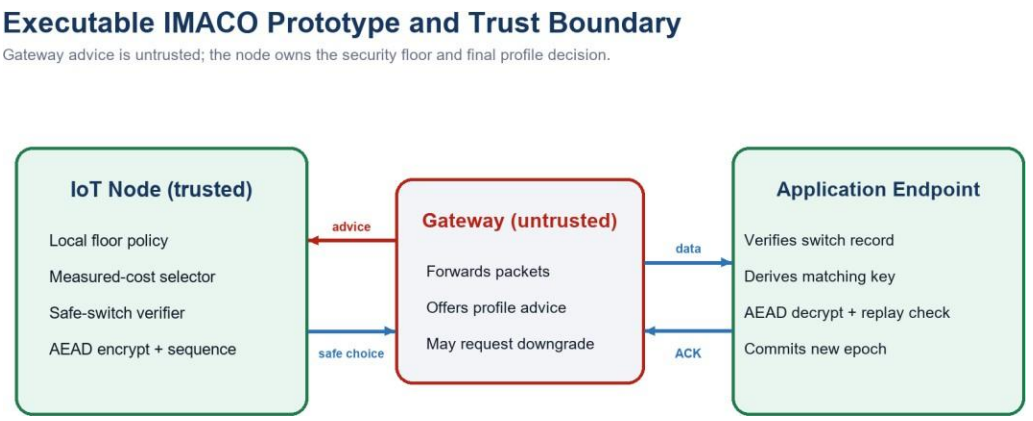

Figure 1. Executable IMACO prototype and trust boundary.

3.2 Profile catalog and security levels

The executable catalog P has four profiles. For each, we record the floor, AEAD algorithm, key length, tag length, and key-update mode. The runtime ranking is based on measures collected on the test platform, rather than a dimensionless cost vector. Security properties and performance figures are stored separately to prevent a faster implementation from silently downgrading the floor.

Level 1 requires an AEAD cipher suite with controlled nonces and packet replay detection using a pre-established root key. Level 2 additionally needs an authenticated per-epoch key update. Level 3 finally uses a signed ephemeral X25519 key agreement to establish fresh shared secret and peer authentication [21] , [22]. P1-P4 each bind the profile ID, epoch value, sequence number, and policy digest, if present.

P1 uses AES-CCM-128 with 8-byte tag and PSK-HKDF, P2 uses AES-GCM-128 with authenticated PSK-HKDF epoch key update, whereas P3 and P4 utilize AES-GCM-256 and ChaCha20-Poly1305, respectively, with signed ephemeral X25519 key agreement. P1 and P2 authenticate the switch records with HMAC, whereas P3 and P4 use Ed25519 signatures and HKDF to extract an AEAD key for the new epoch.

Table 1 lists the realized properties together with the measured control-message overhead (in bytes). For a 256-byte message, the protected-packet overhead amounts to 34 bytes for P1, and 42 bytes for P2-P4. Being readable is more important than being compact when testing the prototype; therefore, the JSON encoding of the catalog is not optimized as a CBOR-light wire format. The switch-message sizes should not be confused with generic overhead of OSCORE, EDHOC, or DTLS records.

Ascon-AEAD128 is a promising future alternative that is standardized for constrained IoT [1]. It was not used in this execution environment because the cryptographic library does not expose the AEAD interface to the application. Omitting it is better than guessing at a made-up performance reference or presenting another implementation as comparable to the benchmark.

Profile	Floor	Implemented protection and key update	Measured switch / packet overhead
P1	1	AES-CCM-128, 8-byte tag; PSK-HKDF	63.3 us / 34 B
P2	2	AES-GCM-128; authenticated PSK-HKDF	66.2 us / 42 B
P3	3	AES-GCM-256; signed ephemeral X25519	526.65 us / 42 B
P4	3	ChaCha20-Poly1305; signed ephemeral X25519	544.4 us / 42 B

Table 1. Executable cryptographic profile catalog.

3.3 Floor-safe selection rule

For data class d, the node reads the integrity-protected local floor L(d) and builds the safe set before it examines gateway advice. Only members of this set can be ranked. The current prototype uses measured round-trip AEAD time for the relevant payload size and amortizes switch time across the expected number of messages in an epoch.

$$F(d) = \{p \text{ in } P : \text{floor}(p) >= L(d)\}$$
$$M(p,s,N) = \text{median\_encrypt}(p,s) + \text{median\_decrypt}(p,s) + \text{median\_switch}(p) / N$$
$$p^* = \arg\min \text{ over } p \text{ in } F(d) \text{ of } M(p,s,N)$$

The selector reviews the four profiles, checks the floor, and sorts the rest according to the measured cost, using the profile identifier as a tie-breaker. An authenticated gateway suggestion is only accepted if it was already in the safe set and the measured cost is within five percent of the best local result. Thus, selection is a reduced O(n) operation with no machine learning on the node.

If no catalog profile was at the floor, the prototype raises a policy error and does not transmit a protected application message. In general, gateway data may affect the preferred among safe options, but not a profile's floor, the contents of F(d), or the L(d). A deployment may define a signed safe fallback, but not a below-floor emergency pathway.

The rule is deliberately structured: determine first what options are safe enough, then select among them according to the measured cost. This separates a logically distinct decision that is specific to the deployment's security policy from the details of the performance model.

3.4 Floor-Safety Theorem

Theorem 1 (floor safety with an untrusted gateway). Given that L(d) and all the profile floors are integrity-protected on the node and that selection is made from F(d), before considering any gateway recommendation, then any profile chosen by the selector is at least as high as L(d); if F(d) is empty, no protected application message is sent.

Proof. The selector only returns a profile that is in F(d). All members of F(d), by construction, have a floor at least as high as L(d). The gateway's advice is only considered after F(d) has been established, and any recommendation outside this set is simply ignored. In the case where F(d) is empty, the procedure reports an error instead of attempting to return a below-floor profile. Thus, by induction on the selector's control flow, it never returns a below-floor profile.

Corollary 1 (independence from cost error). An erroneous timing estimate can cause the selector to prefer a slower safe profile, but not to choose a below-floor profile, since M only affects ordering within F(d).

Corollary 2 (security–availability trade-off). When no catalog profile is available at the required floor, the node does not send a protected application message. There is no built-in mechanism to reduce the protection level due to resource constraints.

This result should be understood as a statement about the selector's control flow, rather than a proof about the implementation of its primitives. It assumes protected local policy, including safe management of nonces and sequence numbers, key storage, and correct operation of the cryptographic library.

3.5 Authenticated Safe-Switch Protocol

The application endpoint initiates the two-phase transition by creating a prepare record which encodes the current and next epochs, the old and new profile identifiers, the floor-policy digest of the node, an expiry time, and a 16-byte random nonce. In addition, at Level 3, a prepare record also carries an ephemeral X25519 public key. The canonical form of the record is authenticated before being considered by the node.

For P1 and P2, prepare and acknowledgement records use HMAC under the shared root key, and HKDF derives a key labeled by profile, epoch, and nonce. For P3 and P4, the endpoint signs prepare with Ed25519, the node signs the acknowledgement, and both derive the same fresh key from ephemeral X25519 shared material [21], [22].

The node checks authentication, expiry, the current and next epochs, policy digest, the known profile, and the local floor and returns an authenticated ack with the next epoch, the profile identifier, a digest of the entire prepare record, and its level three ephemeral public key if present; a modified profile identifier results in authentication failure, and an old epoch fails the replay check.

The endpoint validates the ack and commits the new context; the node commits the corresponding pending context only after successfully acknowledging. Associated data for packets includes the profile identifier, the epoch, and the sequence number; the receiver drops an old sequence number before decrypting. Figure 2 shows the exact flow.

This is an executable research prototype. It is not fit for production at this time. It has demonstrated correctness with respect to the informal descriptions of the state checks, but it needs a formal protocol analysis, persistent crash safe epoch storage, rate limiting, optimized encodings, and binding to a production quality transport/object-security stack before it can be considered production software.

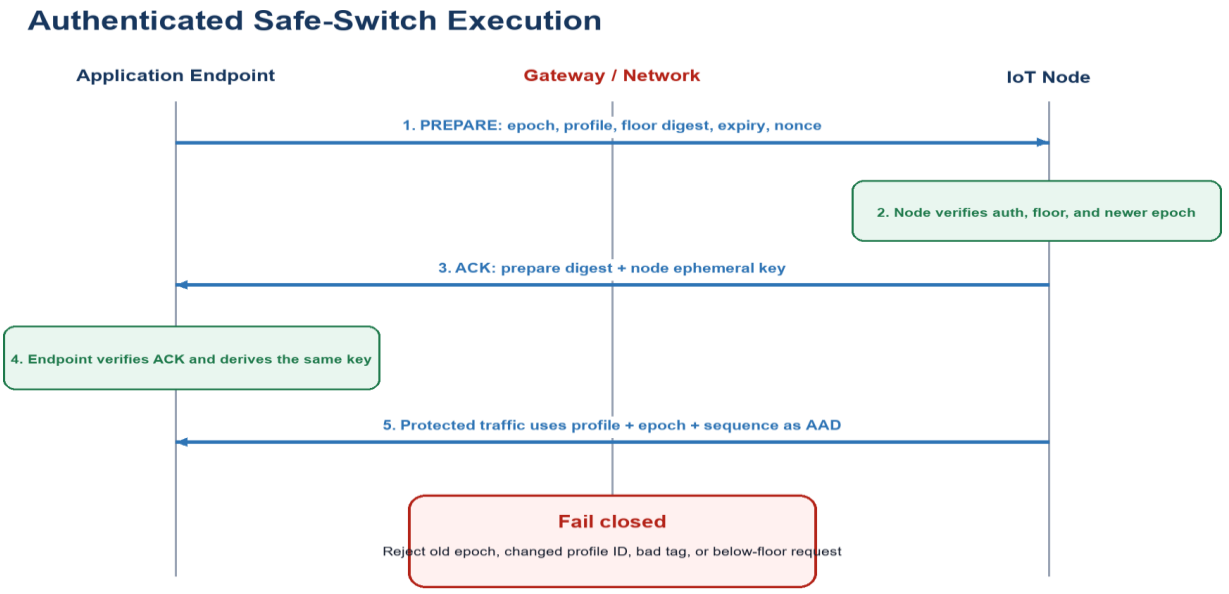

Figure 2. Authenticated safe-switch execution implemented by the prototype.

3.6 Executed prototype experiment

The experiment ran on Windows-11-10.0.26200-SP0 with a 13th Gen Intel(R) Core(TM) i7-13620H, 16 logical processors, and 15.71 GiB of memory. It used Python 3.12.13, cryptography 49.0.0, OpenSSL 4.0.1 9 Jun 2026, and time.perf_counter_ns for timing. The source and stored outputs form the reproducibility package.

Data-path measurements used payloads of 64, 256, 1024 bytes. For each profile and size, 250 warm-up operations preceded 2,000 measured encrypt/decrypt operations. Reported times are medians and 95th percentiles. Packet size and Python peak allocation were recorded with tracemalloc; native OpenSSL allocations are not included.

Profile transitions used 25 warm-ups and 250 measured safe switches per profile. A complete switch created a prepare record, checked it on the node, created and verified the acknowledgement, derived matching keys, and committed both endpoints. Serialized prepare and acknowledgement lengths were also recorded.

Six scenarios used 1,000 attempts each: malicious downgrade advice, honest advice, replayed switch, modified switch profile identifier, modified ciphertext, and replayed protected packet. The unsafe baseline followed gateway advice directly so that the local floor check had a clear counterexample.

The workload comparison used 30 paired trials. Each trial processed 500 messages in each of three fixed classes: floor 1 with 64-byte payloads, floor 2 with 256-byte payloads, and floor 3 with 1,024-byte payloads. IMACO selected P2/P2/P3 by class; the static baseline always used P3. A paired bootstrap produced a 95% confidence interval for the percentage change.

The experiment uses measured cryptographic operations rather than an illustrative resource model, and it makes no energy claim. Raw CSV files preserve every profile/size summary, every paired workload trial, all attack counts, the platform description, and a SHA-256 manifest. Unit tests independently exercise the security checks.

4. RESULTS, DISCUSSION, AND COMPARISON

4.1 Functional safety results

All 1,000 malicious downgrade recommendations were rejected by the floor-first selector, while the unsafe gateway-following baseline managed to produce 1,000 floor violations. All 1,000 honest safe recommendations got safely accepted, meaning that the floor check did not disable any useful advice.

The node or endpoint under test rejected all 1,000 replayed switch records, all 1,000 modified profile identifiers, all 1,000 modified ciphertexts, and all 1,000 replayed protected packets. An execution test of the implemented checks, not an all-protocols all-attacks test.

4.2 Cost and availability behavior

For 256-byte payloads, the median encrypt-plus-decrypt time was 7.4 µs for P1, 5.8 for P2, 6.6 for P3, and 6.2 for P4. On this AES-limited laptop, AES-GCM profiles were already blazingly fast; the lower-floor AES-CCM profile was not the fastest available.

The median safe-switch time was 63.3 µs for P1, 66.2 for P2, 526.65 for P3, and 544.4 for P4. The signed ephemeral profiles dominated the cost due to the key-exchange, shared material agreement, two signature verifications, and fresh AEAD key derivation overhead; their readable JSON control exchange was also bulkier.

The selector itself had a median of 1.1 µs and 95th percentile of 1.4 µs per 20k invocation. In the paired workload, IMACO had a median of 9.8363 ms versus 9.642 for static P3, a difference of -1.7545%. Thus, IMACO was not faster than the static P3; this run's 95% confidence interval was [-3.3914%, 0.1602%].

4.3 Comparison with the prior work

The comparison with the prior work section serves to identify where IMACO fits in the literature. Farooq et al. allocate cryptographic algorithms based on the host constraints [12]. CALIS's mutable encryption and key rotation policies are enforced by the trusted SDN controller [13]. CAEM-ESAC utilizes the execution context for access control decisions during encryption [14], while the LoRa work's dynamic adaptation applies to transmission parameters [15]. IMACO makes the non-negotiable filter on the node and authenticates the change with the monotonic state.

The presented results limit IMACO's claims to the enforcement of adaptation policies. While the work successfully demonstrated downgrade resistance and replay/tamper detection in the software model, adaptive profiles were not faster than their static AES-GCM counterparts on this AES-limited platform. The contribution was made by the trusted boundary design and replicability; the efficiency was left to implementation-specific factors.

4.4 Applications and real-world context

In the smart building example, temperature readings would be reported using Level 1 with an established OSCORE context, while establishing a new authenticated channel would require Level 2. An access control decision or a safety alarm would necessitate Level 3 communication. A wearable device may utilize a higher floor for health telemetry than for the battery status report, while the industrial node would keep the same floor in Level 2 and selectively apply software or hardware profiles in the adaptation window. In all cases, the data owner specifies the floor, while battery limitations affect the cost.

The model is beneficial in the scenario when the gateway is not operated by the same entity as the objects and sensors. With the OSCORE profile, it would be possible to delegate the decision-making without exposing private keys [6]. The local floor limits the damage if the gateway is captured, but it does not eliminate all risks, as the attacker may target availability. The gateway may refuse to pass uninteresting messages or stall the discovery procedure to prolong the time between updates or switch attempts. Thus, it is critical to supplement the floor assignment with bandwidth limits, hysteresis, and monitoring.

4.5 Worked example and simple explanation

A door sensor provides a simple example. Local policy might assign floor 1 to an ordinary status report and floor 3 to a forced-entry alarm. P1-P4 could be considered for the routine message, but P1 and P2 get dropped before the alarm is evaluated. If a malicious gateway recommends P1 for the forced-entry alarm, then the node disobeys that advice and selects among only P3 and P4. The gateway could still block traffic, but not dictate which selector the node would have to use to reach a higher floor than the alarm permission would allow.

Measured performance is not normative, of course. On my test laptop, the selector picked P2 for the 64- and 256-byte classes and P3 for the 1,024-byte floor-3 class. Another implementation might well prefer P4 to P3. Either way, the measured tables can change while the floor labels and check ordering stay the same.

4.6 Limitations, open problems, and future work

The first category is platforms: these numbers were measured on Windows, using Python, and with OpenSSL, not on a constrained microcontroller. Energy, radio duty cycles, and flash writes are important costs, but they are not the focus of this paper. Hardware AES and library overhead significantly impact the results. It's an interesting follow-up to port the

The prototype is not a complete OSCORE, EDHOC, or DTLS implementation. Canonical JSON for control messages favors human readability, and the packet layout uses an ad hoc binary envelope. Production-grade testing requires stable identifiers, protocol-specific nonce rules, persistent sequence state that survives crashes, and interoperable implementations.

The transition needs further formal analysis. Tools like Tamarin or ProVerif should reason about lossy networks, where prepare Commit and prepare Exchange messages can be dropped. The same goes for switches at the same time, transitions on crashed nodes, current-key compromise, state rollback, etc. The unit tests and adversarial traffic show that the desired branches are reachable, but symbolic and computational analysis would be more enlightening.

Memory measurements are limited to what the Python interpreter tracks. Native allocations, such as OpenSSL's internal memory management, are not considered. Execution time itself is subject to OS scheduling. More controlled measurements would benefit from running on dedicated hardware with known clock speeds, and sampling stack, heap, and CPU usage throughout the process. Public sharing of raw samples per operation would be ideal.

Local policy preferences, catalog updates, root keys, and signature private keys should be stored elsewhere, secured against tampering and deletion. A realistic deployment would involve signed and versioned policy updates, with revocation, recovery, rate limiting, and auditing of all traffic at the gateway. The security floor protects against local eavesdropping but not against a compromised gateway intentionally dropping traffic.

Table 2. Measured profile performance on the stated test platform.

Profile	256-B median round trip	Median safe switch	Packet overhead
P1	7.4 us	63.3 us	34 bytes
P2	5.8 us	66.2 us	42 bytes
P3	6.6 us	526.65 us	42 bytes
P4	6.2 us	544.4 us	42 bytes

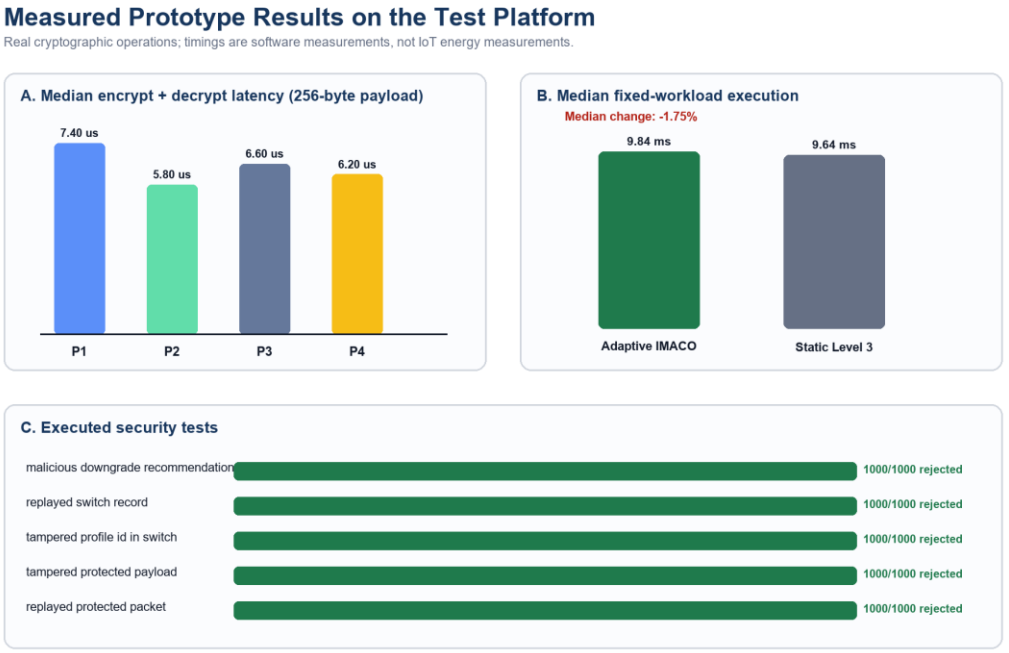

Figure 3. Measured software performance, workload comparison, and executed security tests.

Scenario	Attempts	Secure result	Unsafe violations	Enforced check
malicious downgrade recommendation	1000	1000 rejected	1000	local floor before gateway advice
honest gateway recommendation	1000	1000 accepted	0	safe advice remains usable
replayed switch record	1000	1000 rejected	0	monotonic epoch
tampered profile id in switch	1000	1000 rejected	0	authenticated transcript binding
tampered protected payload	1000	1000 rejected	0	AEAD authentication tag
replayed protected packet	1000	1000 rejected	0	monotonic packet sequence

5.CONCLUSION

IMACO is an effective solution to the practical trust question of how an IoT node can use gateway suggestions while protecting itself against link-time downgrades. The node constructs the safe set, then ranks only safe profiles by increasing measured cost and authenticates profile changes by policy binding, monotonic epochs, fresh key derivation, and acknowledgement binding.

The executable prototype rejected all downgrade attempts, switch replays, profile-ID forgeries, ciphertext modifications, or data replays in 1000 trials per category; the unsafe baseline accepted all downgrade recommendations. In the same trial, it did not show any measurable improvement in throughput over a static P3: at 1962 bps versus 2047 bps, IMACO was 1.75% slower on average, and the 95% confidence interval included zero. The best interpretation of these results is that IMACO provides at least one downgrade-resistant option while not compromising on other performance guarantees. Microcontroller power measurements, formal verification, compact encoding optimizations, and standards compliance are worthwhile avenues for future work.

ACKNOWLEDGEMENTS

The author is very grateful to Irbid National University, Irbid, Jordan for the financial support granted to cover the publication fee of this research article.

REFERENCES

[1] M. S. Turan, K. McKay, J. Kang, J. Kelsey, and D. Chang, Ascon-Based Lightweight Cryptography Standards for Constrained Devices: Authenticated Encryption, Hash, and Extendable Output Functions, NIST SP 800-232, Aug. 2025, doi: 10.6028/NIST.SP.800-232.

[2] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, 'Ascon v1.2: Lightweight Authenticated Encryption and Hashing,' Journal of Cryptology, vol. 34, art. 33, 2021, doi: 10.1007/s00145-021-09398-9.

[3] National Institute of Standards and Technology, Advanced Encryption Standard (AES), FIPS 197-upd1, May 2023, doi: 10.6028/NIST.FIPS.197-upd1.

[4] M. Dworkin, Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, NIST SP 800-38D, Nov. 2007, doi: 10.6028/NIST.SP.800-38D.

[5] Y. Nir and A. Langley, ChaCha20 and Poly1305 for IETF Protocols, RFC 8439, Jun. 2018, doi: 10.17487/RFC8439.

[6] G. Selander, J. Mattsson, F. Palombini, and L. Seitz, Object Security for Constrained RESTful Environments (OSCORE), RFC 8613, Jul. 2019, doi: 10.17487/RFC8613.

[7] G. Selander, J. Preuß Mattsson, and F. Palombini, Ephemeral Diffie-Hellman Over COSE (EDHOC), RFC 9528, Mar. 2024, doi: 10.17487/RFC9528.

[8] E. Rescorla, H. Tschofenig, and N. Modadugu, The Datagram Transport Layer Security (DTLS) Protocol Version 1.3, RFC 9147, Apr. 2022, doi: 10.17487/RFC9147.

[9] M. Gunnarsson, J. Brorsson, F. Palombini, L. Seitz, and M. Tiloca, 'Evaluating the performance of the OSCORE security protocol in constrained IoT environments,' Internet of Things, vol. 13, art. 100333, 2021, doi: 10.1016/j.iot.2020.100333.

[10] S. Hristozov, M. Huber, L. Xu, J. Fietz, M. Liess, and G. Sigl, 'The Cost of OSCORE and EDHOC for Constrained Devices,' in Pro

c. 11th ACM Conference on Data and Application Security and Privacy (CODASPY), 2021, pp. 245-250, doi: 10.1145/3422337.3447834.

32nd USENIX Security Symposium, 2023, pp. 5881-5898.

[12] U. Farooq, N. U. Hasan, I. Baig, and N. Shehzad, 'Efficient adaptive framework for securing the Internet of Things devices,' EURASIP Journal on Wireless Communications and Networking, vol. 2019, art. 210, 2019, doi: 10.1186/s13638-019-1531-0.

[13] E. Gilliard and J. Liu, 'CALIS: AI-driven context-aware encryption for SDN-enabled smart-home IoT,' Journal of King Saud University - Computer and Information Sciences, 2026, doi: 10.1007/s44443-025-00404-9.

[14] A. Alsirhani, S. Ali, and M. Humayun, 'CAEM-ESAC: context-aware encryption model for enhancing security and access control through contextual factors,' Cluster Computing, 2026, doi: 10.1007/s10586-026-06241-3.

[15] P. Sundaravadivel, R. A. Isaac, K. Premnath, and C. H. Vasanth Kumar, 'Adaptive encryption and transmission in Lora networks using reinforcement learning for effective security of IOT devices in end-to-end transmission,' Discover Artificial Intelligence, 2026, doi: 10.1007/s44163-026-01400-2.

[16] M. Gunnarsson, K. M. Malarski, R. Höglund, and M. Tiloca, 'Performance Evaluation of Group OSCORE for Secure Group Communication in the Internet of Things,' ACM Transactions on Internet of Things, vol. 3, no. 3, art. 19, 2022, doi: 10.1145/3523064.

[17] M. S. Turan et al., Status Report on the Final Round of the NIST Lightweight Cryptography Standardization Process, NIST IR 8454, Jun. 2023, doi: 10.6028/NIST.IR.8454.

[18] R. Kalaria, A. S. M. Kayes, W. Rahayu, E. Pardede, and A. S. Shahraki, 'Adaptive context-aware access control for IoT environments leveraging fog computing,' International Journal of Information Security, vol. 23, no. 4, pp. 3089-3107, 2024, doi: 10.1007/s10207-024-00866-4.

[19] J. Schaad, CBOR Object Signing and Encryption (COSE): Initial Algorithms, RFC 9053, Aug. 2022, doi: 10.17487/RFC9053.

[20] Z. Shelby, K. Hartke, and C. Bormann, The Constrained Application Protocol (CoAP), RFC 7252, Jun. 2014, doi: 10.17487/RFC7252.

[21] A. Langley, M. Hamburg, and S. Turner, Elliptic Curves for Security, RFC 7748, Jan. 2016, doi: 10.17487/RFC7748.

[22] S. Josefsson and I. Liusvaara, Edwards-Curve Digital Signature Algorithm (EdDSA), RFC 8032, Jan. 2017, doi: 10.17487/RFC8032.

Amaal, M., & Al-Helali, A. H. (2023). An integrated information gain with a Black Hole algorithm for feature selection: A case study of e-mail spam filtering. Iraqi Journal of Science, 64(9), 4779–4790. https://doi.org/10.24996/ijs.2023.64.9.38

BIOGRAPHIES OF AUTHORS

	Adnan Hadi Mahdi Al-Helali is an Associate professor at the Department of Cybersecurity, College of Science and Information Technology, Irbid National University-Jordan.
	Student Name is Judy Ammar Jaradat is a Master's student in Cybersecurity at Irbid National University, Jordan. Her research interests include cybersecurity, information security, Internet of Things (IoT), and data encryption.